
pyFlowCL

Versión 1.0.5

mariofix

18 de enero de 2022

Contents:

1. pyflowcl	1
1.1. pyflowcl package	1
2. Indices and tables	11
Índice de Módulos Python	13
Índice	15

1.1 pyflowcl package

1.1.1 Submodules

1.1.2 pyflowcl.Clients module

pyflowcl.Clients

Este modulo implementa el Cliente API genérico.

```
class pyflowcl.Clients.ApiClient(api_url: str = 'https://sandbox.flow.cl/api', api_key: str = '', api_secret: str = '')
```

Bases: object

Clase ApiClient con los objetos para realizar llamadas

Implementa todos los métodos de dataclass.

se instancia con:

```
cliente = ApiClient(api_url, api_key, api_secret)
```

el cliente luego debe ser entregado como primer parametro de las clases incorporadas:

```
pay = Payment.create(cliente, [...])
```

```
api_key: str = ''
```

```
api_secret: str = ''
```

```
api_url: str = 'https://sandbox.flow.cl/api'
```

```
get(url: str, query_string: Dict[str, Any]) → Dict[str, Any]
```

Reimplementa get

Tipo del valor devuelto dict

make_signature(*params: Dict[str, Any]*) → str

Crea el Hash de validacion para ser enviado con la informacion

Tipo del valor devuelto str

post(*url: str, post_data: Dict[str, Any]*) → Dict[str, Any]

Reimplementa post

Tipo del valor devuelto dict

put(*url: str, put_data: Dict[str, Any]*) → Dict[str, Any]

Reimplementa put

Tipo del valor devuelto dict

1.1.3 pyflowcl.Payment module

pyflowcl.Payment.create(*apiclient: pyflowcl.Clients.ApiClient, payment_data: Dict[str, Any]*) →

pyflowcl.models.PaymentResponse

Este método permite crear una orden de pago a Flow y recibe como respuesta la URL para redirigir el browser del pagador y el token que identifica la transacción. La url de redirección se debe formar concatenando los valores recibidos en la respuesta de la siguiente forma:

url + «?token=» +token

Una vez que el pagador efectúe el pago, Flow notificará el resultado a la página del comercio que se envió en el parámetro urlConfirmation.

pyflowcl.Payment.createEmail(*apiclient: pyflowcl.Clients.ApiClient, payment_data: Dict[str, Any]*) →

pyflowcl.models.PaymentResponse

Permite generar un cobro por email. Flow emite un email al pagador que contiene la información de la Orden de pago y el link de pago correspondiente. Una vez que el pagador efectúe el pago, Flow notificará el resultado a la página del comercio que se envió en el parámetro urlConfirmation.

pyflowcl.Payment.getPayments(*apiclient: pyflowcl.Clients.ApiClient, payment_info: Dict[str, Any]*) →

pyflowcl.models.PaymentList

Este método permite obtener la lista paginada de pagos recibidos en un día. Los objetos pagos de la lista tienen la misma estructura de los retornados en los servicios payment/getStatus

pyflowcl.Payment.getStatus(*apiclient: pyflowcl.Clients.ApiClient, token: str*) →

pyflowcl.models.PaymentStatus

Obtiene el estado de un pago previamente creado, el parametro token hace referencia a notification id, el cual se recibe luego de procesado un pago

pyflowcl.Payment.getStatusByCommerceId(*apiclient: pyflowcl.Clients.ApiClient, commerceId: str*) →

pyflowcl.models.PaymentStatus

Obtiene el estado de un pago previamente creado, el parametro token hace referencia a notification id, el cual se recibe luego de procesado un pago

pyflowcl.Payment.getStatusByFlowOrder(*apiclient: pyflowcl.Clients.ApiClient, flowOrder: int*) →

pyflowcl.models.PaymentStatus

Obtiene el estado de un pago previamente creado, el parametro token hace referencia a notification id, el cual se recibe luego de procesado un pago

1.1.4 pyflowcl.Refund module

`pyflowcl.Refund.create`(*apiclient*: `pyflowcl.Clients.ApiClient`, *refund_data*: `Dict[str, Any]`) → `pyflowcl.models.RefundStatus`

Este servicio permite crear una orden de reembolso. Una vez que el receptor del reembolso acepte o rechaze el reembolso, Flow notificará vía POST a la página del comercio identificada en `urlCallback` pasando como parámetro `token`

En esta página, el comercio debe invocar el servicio `refund/getStatus` para obtener el estado del reembolso.

`pyflowcl.Refund.getStatus`(*apiclient*: `pyflowcl.Clients.ApiClient`, *token*: `str`) → `pyflowcl.models.RefundStatus`
Permite obtener el estado de un reembolso solicitado. Este servicio se debe invocar desde la página del comercio que se señaló en el parámetro `urlCallback` del servicio `refund/create`.

1.1.5 pyflowcl.models module

pyflowcl.models

Modelos de distintos objetos del paquete

```
class pyflowcl.models.BatchCollectRejectedRow(customerId: 'Optional[str]' = None, commerceOrder:
                                                'Optional[str]' = None, rowNum: 'Optional[int]' =
                                                None, parameter: 'Optional[str]' = None, errorCode:
                                                'Optional[int]' = None, errorMsg: 'Optional[str]' =
                                                None)
```

Bases: `object`

`commerceOrder`: `Optional[str]` = `None`

`customerId`: `Optional[str]` = `None`

`errorCode`: `Optional[int]` = `None`

`errorMsg`: `Optional[str]` = `None`

`static from_dict`(*d*: `Dict[str, Any]`) → `pyflowcl.models.BatchCollectRejectedRow`

`parameter`: `Optional[str]` = `None`

`rowNum`: `Optional[int]` = `None`

```
class pyflowcl.models.BatchCollectRequest(apiKey: 'str' = 'API_KEY', batchRows: 'str' = '', byEmail: 'int'
                                           = 0, forward_days_after: 'Optional[int]' = None,
                                           forward_times: 'Optional[int]' = None, timeout:
                                           'Optional[int]' = None, urlCallBack: 'str' = '',
                                           urlConfirmation: 'str' = '', urlReturn: 'str' = '', s: 'str' = '')
```

Bases: `object`

`apiKey`: `str` = `'API_KEY'`

`batchRows`: `str` = `''`

`byEmail`: `int` = `0`

`forward_days_after`: `Optional[int]` = `None`

`forward_times`: `Optional[int]` = `None`

`static from_dict`(*d*: `Dict[str, Any]`) → `pyflowcl.models.BatchCollectRequest`

`s`: `str` = `''`

```
    timeout: Optional[int] = None
    urlCallBack: str = ''
    urlConfirmation: str = ''
    urlReturn: str = ''

class pyflowcl.models.BatchCollectResponse(token: 'Optional[str]' = None, receivedRows: 'Optional[int]'
                                           = None, acceptedRows: 'Optional[int]' = None,
                                           rejectedRows: 'Optional[List[BatchCollectRejectedRow]]' =
                                           None)

    Bases: object

    acceptedRows: Optional[int] = None
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.BatchCollectResponse
    receivedRows: Optional[int] = None
    rejectedRows: Optional[List[pyflowcl.models.BatchCollectRejectedRow]] = None
    token: Optional[str] = None

class pyflowcl.models.CollectObject(customer_id: str, commerce_order: str, subject: str, amount: float,
                                    currency: Optional[str] = None, payment_method: Optional[float] =
                                    None, optional: Optional[str] = None)

    Bases: object

    Objeto de cobro para un lote de cobros

    amount: float
    commerce_order: str
    currency: Optional[str] = None
    customer_id: str
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.CollectObject
    optional: Optional[str] = None
    payment_method: Optional[float] = None
    subject: str

class pyflowcl.models.CollectRequest(amount: float = 0, apiKey: str = 'API_KEY', byEmail: Optional[int]
                                    = None, commerceOrder: str = "", currency: Optional[str] = None,
                                    customerId: str = "", forward_days_after: Optional[int] = None,
                                    forward_times: Optional[int] = None, ignore_auto_charging:
                                    Optional[int] = None, merchantId: Optional[str] = None, optional:
                                    Optional[str] = None, paymentMethod: Optional[int] = 9, subject:
                                    Optional[str] = None, timeout: Optional[int] = None,
                                    urlConfirmation: str = "", urlReturn: str = "", s: str = "")

    Bases: object

    Objeto para generar un correo electronico de pago

    amount: float = 0
    apiKey: str = 'API_KEY'
    byEmail: Optional[int] = None
    commerceOrder: str = ''
```



```

currency: Optional[str] = None
customerId: str = ''
forward_days_after: Optional[int] = None
forward_times: Optional[int] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.CollectRequest
ignore_auto_charging: Optional[int] = None
merchantId: Optional[str] = None
optionals: Optional[str] = None
paymentMethod: Optional[int] = 9
s: str = ''
subject: Optional[str] = None
timeout: Optional[int] = None
urlConfirmation: str = ''
urlReturn: str = ''

class pyflowcl.models.CollectResponse(commerce_order: Optional[str] = None, flow_order:
    Optional[float] = None, paymen_result:
    Optional[pyflowcl.models.PaymentStatus] = None, status:
    Optional[int] = None, token: Optional[str] = None, type:
    Optional[float] = None, url: Optional[str] = None)

Bases: object
Objeto para CollectResponse
commerce_order: Optional[str] = None
flow_order: Optional[float] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.CollectResponse
paymen_result: Optional[pyflowcl.models.PaymentStatus] = None
status: Optional[int] = None
token: Optional[str] = None
type: Optional[float] = None
url: Optional[str] = None

class pyflowcl.models.Customer(created: str = "", creditCardType: Optional[str] = None, customerId: str = "",
    email: str = "", externalId: Optional[str] = None, last4CardDigits:
    Optional[str] = None, name: str = "", pay_mode: Optional[str] = None,
    registerDate: Optional[str] = None, status: int = 0)

Bases: object
Customer Object
created: str = ''
creditCardType: Optional[str] = None
customerId: str = ''
email: str = ''

```

```
externalId: Optional[str] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.Customer
last4CardDigits: Optional[str] = None
name: str = ''
pay_mode: Optional[str] = None
registerDate: Optional[str] = None
status: int = 0
```

class pyflowcl.models.CustomerChargeRequest(*amount: float = 0, apiKey: str = 'API_KEY', commerceOrder: str = "", currency: Optional[str] = None, optionals: Optional[str] = None, subject: str = "", s: str = ""*)

Bases: object

Objeto para generar una URL de pago

```
amount: float = 0
apiKey: str = 'API_KEY'
commerceOrder: str = ''
currency: Optional[str] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.CustomerChargeRequest
optionals: Optional[str] = None
s: str = ''
subject: str = ''
```

class pyflowcl.models.CustomerList(*total: Optional[float] = None, hasMore: Optional[bool] = None, data: Optional[List[Dict[Any, Any]]] = None*)

Bases: object

Lista de Clientes

```
data: Optional[List[Dict[Any, Any]]] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.CustomerList
hasMore: Optional[bool] = None
total: Optional[float] = None
```

class pyflowcl.models.CustomerRegisterResponse(*url: Optional[str] = None, token: Optional[str] = None*)

Bases: object

Objeto respuesta

```
static from_dict(d: Dict[str, Any]) → pyflowcl.models.CustomerRegisterResponse
token: Optional[str] = None
url: Optional[str] = None
```

class pyflowcl.models.CustomerRegisterStatusResponse(*creditCardType: str = "", customerId: str = "", last4CardDigits: str = "", status: int = 0*)

Bases: object

Objeto respuesta

```

    creditCardType: str = ''
    customerId: str = ''
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.CustomerRegisterStatusResponse
    last4CardDigits: str = ''
    status: int = 0
class pyflowcl.models.CustomerRequest(apiKey: str = "", customerId: str = "", email: str = "", externalId: str
                                     = "", name: str = "", s: str = "")
    Bases: object
    CustomerRequest Object
    apiKey: str = ''
    customerId: str = ''
    email: str = ''
    externalId: str = ''
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.CustomerRequest
    name: str = ''
    s: str = ''
exception pyflowcl.models.GenericError(data)
    Bases: BaseException
class pyflowcl.models.PaymentList(total: Optional[float] = None, hasMore: Optional[bool] = None, data:
                                   Optional[List[Dict[Any, Any]]] = None)
    Bases: object
    Lista de pagos
    data: Optional[List[Dict[Any, Any]]] = None
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.PaymentList
    hasMore: Optional[bool] = None
    total: Optional[float] = None
class pyflowcl.models.PaymentRequest(amount: float = 0, apiKey: str = 'API_KEY', commerceOrder: str =
                                     ", currency: Optional[str] = None, email: str = 'correo@ejemplo.cl',
                                     merchantId: Optional[str] = None, optional: Optional[str] = None,
                                     payment_currency: str = 'CLP', payment_method: Optional[int] =
                                     None, subject: str = "", timeout: Optional[int] = None,
                                     urlConfirmation: str = "", urlReturn: str = "", s: str = "")
    Bases: object
    Objeto para generar una URL de pago
    amount: float = 0
    apiKey: str = 'API_KEY'
    commerceOrder: str = ''
    currency: Optional[str] = None
    email: str = 'correo@ejemplo.cl'
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.PaymentRequest

```

```
merchantId: Optional[str] = None
optional: Optional[str] = None
payment_currency: str = 'CLP'
payment_method: Optional[int] = None
s: str = ''
subject: str = ''
timeout: Optional[int] = None
urlConfirmation: str = ''
urlReturn: str = ''
```

```
class pyflowcl.models.PaymentRequestEmail(amount: float = 0, apiKey: str = 'API_KEY', commerceOrder:
                                         str = "", currency: Optional[str] = None, email: str =
                                         'correo@ejemplo.cl', forward_days_after: Optional[int] =
                                         None, forward_times: Optional[int] = None, merchantId:
                                         Optional[str] = None, optional: Optional[str] = None,
                                         payment_currency: Optional[str] = None, subject:
                                         Optional[str] = None, timeout: Optional[int] = None,
                                         urlConfirmation: str = "", urlReturn: str = "", s: str = "")
```

Bases: object

Objeto para generar un correo electronico de pago

```
amount: float = 0
apiKey: str = 'API_KEY'
commerceOrder: str = ''
currency: Optional[str] = None
email: str = 'correo@ejemplo.cl'
forward_days_after: Optional[int] = None
forward_times: Optional[int] = None
static from_dict(d: Dict[str, Any]) → pyflowcl.models.PaymentRequestEmail
merchantId: Optional[str] = None
optional: Optional[str] = None
payment_currency: Optional[str] = None
s: str = ''
subject: Optional[str] = None
timeout: Optional[int] = None
urlConfirmation: str = ''
urlReturn: str = ''
```

```
class pyflowcl.models.PaymentResponse(url: Optional[str] = None, token: Optional[str] = None, flowOrder:
                                       Optional[float] = None)
```

Bases: object

Objeto respuesta de una creacion de pago

```

    flowOrder: Optional[float] = None
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.PaymentResponse
    token: Optional[str] = None
    url: Optional[str] = None
class pyflowcl.models.PaymentStatus(flow_order: Optional[int] = None, commerce_order: Optional[str] =
    None, request_date: Optional[str] = None, status: Optional[int] =
    None, subject: Optional[str] = None, currency: Optional[str] = None,
    amount: Optional[float] = None, payer: Optional[str] = None,
    optional: Optional[str] = None, pending_info: Optional[Dict[Any,
    Any]] = None, payment_data: Optional[Dict[Any, Any]] = None,
    merchant_id: Optional[str] = None)

```

Bases: object

Objeto para obtener el estado de un pago

```

    amount: Optional[float] = None
    commerce_order: Optional[str] = None
    currency: Optional[str] = None
    flow_order: Optional[int] = None
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.PaymentStatus
    merchant_id: Optional[str] = None
    optional: Optional[str] = None
    payer: Optional[str] = None
    payment_data: Optional[Dict[Any, Any]] = None
    pending_info: Optional[Dict[Any, Any]] = None
    request_date: Optional[str] = None
    status: Optional[int] = None
    subject: Optional[str] = None
class pyflowcl.models.RefundRequest(amount: float = 0, apiKey: str = 'API_KEY', commerceTrxId:
    Optional[str] = None, flowTrxId: Optional[float] = None,
    receiverEmail: str = 'correo@ejemplo.cl', refundCommerceOrder: str
    = '', urlCallBack: str = '', s: str = '')

```

Bases: object

Refund Request object

```

    amount: float = 0
    apiKey: str = 'API_KEY'
    commerceTrxId: Optional[str] = None
    flowTrxId: Optional[float] = None
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.RefundRequest
    receiverEmail: str = 'correo@ejemplo.cl'
    refundCommerceOrder: str = ''
    s: str = ''

```

```
urlCallback: str = ''  
class pyflowcl.models.RefundStatus(flowRefundOrder: int = 0, date: str = "", status: str = "", amount: float =  
                                0, fee: float = 0)  
    Bases: object  
    Refund object  
    amount: float = 0  
    date: str = ''  
    fee: float = 0  
    flowRefundOrder: int = 0  
    static from_dict(d: Dict[str, Any]) → pyflowcl.models.RefundStatus  
    status: str = ''
```

1.1.6 Module contents

CAPÍTULO 2

Indices and tables

- `genindex`
- `modindex`

p

- `pyflowcl`, [10](#)
- `pyflowcl.Clients`, [1](#)
- `pyflowcl.models`, [3](#)
- `pyflowcl.Payment`, [2](#)
- `pyflowcl.Refund`, [3](#)

A

acceptedRows (atributo de py-flowcl.models.BatchCollectResponse), 4
amount (atributo de pyflowcl.models.CollectObject), 4
amount (atributo de pyflowcl.models.CollectRequest), 4
amount (atributo de py-flowcl.models.CustomerChargeRequest), 6
amount (atributo de pyflowcl.models.PaymentRequest), 7
amount (atributo de py-flowcl.models.PaymentRequestEmail), 8
amount (atributo de pyflowcl.models.PaymentStatus), 9
amount (atributo de pyflowcl.models.RefundRequest), 9
amount (atributo de pyflowcl.models.RefundStatus), 10
api_key (atributo de pyflowcl.Clients.ApiClient), 1
api_secret (atributo de pyflowcl.Clients.ApiClient), 1
api_url (atributo de pyflowcl.Clients.ApiClient), 1
ApiClient (clase en pyflowcl.Clients), 1
apiKey (atributo de py-flowcl.models.BatchCollectRequest), 3
apiKey (atributo de pyflowcl.models.CollectRequest), 4
apiKey (atributo de py-flowcl.models.CustomerChargeRequest), 6
apiKey (atributo de pyflowcl.models.CustomerRequest), 7
apiKey (atributo de pyflowcl.models.PaymentRequest), 7
apiKey (atributo de py-flowcl.models.PaymentRequestEmail), 8
apiKey (atributo de pyflowcl.models.RefundRequest), 9

B

BatchCollectRejectedRow (clase en pyflowcl.models), 3
BatchCollectRequest (clase en pyflowcl.models), 3
BatchCollectResponse (clase en pyflowcl.models), 4
batchRows (atributo de py-flowcl.models.BatchCollectRequest), 3

byEmail (atributo de py-flowcl.models.BatchCollectRequest), 3
byEmail (atributo de pyflowcl.models.CollectRequest), 4

C

CollectObject (clase en pyflowcl.models), 4
CollectRequest (clase en pyflowcl.models), 4
CollectResponse (clase en pyflowcl.models), 5
commerce_order (atributo de py-flowcl.models.CollectObject), 4
commerce_order (atributo de py-flowcl.models.CollectResponse), 5
commerce_order (atributo de py-flowcl.models.PaymentStatus), 9
commerceOrder (atributo de py-flowcl.models.BatchCollectRejectedRow), 3
commerceOrder (atributo de py-flowcl.models.CollectRequest), 4
commerceOrder (atributo de py-flowcl.models.CustomerChargeRequest), 6
commerceOrder (atributo de py-flowcl.models.PaymentRequest), 7
commerceOrder (atributo de py-flowcl.models.PaymentRequestEmail), 8
commerceTrxId (atributo de py-flowcl.models.RefundRequest), 9
create() (en el módulo pyflowcl.Payment), 2
create() (en el módulo pyflowcl.Refund), 3
created (atributo de pyflowcl.models.Customer), 5
createEmail() (en el módulo pyflowcl.Payment), 2
creditCardType (atributo de py-flowcl.models.Customer), 5
creditCardType (atributo de py-flowcl.models.CustomerRegisterStatusResponse), 6
currency (atributo de pyflowcl.models.CollectObject), 4
currency (atributo de pyflowcl.models.CollectRequest), 4

currency (atributo de <i>py-flowcl.models.CustomerChargeRequest</i>), 6	flow_order (atributo de <i>py-flowcl.models.CollectResponse</i>), 5
currency (atributo de <i>py-flowcl.models.PaymentRequest</i>), 7	flow_order (atributo de <i>py-flowcl.models.PaymentStatus</i>), 9
currency (atributo de <i>py-flowcl.models.PaymentRequestEmail</i>), 8	flowOrder (atributo de <i>py-flowcl.models.PaymentResponse</i>), 8
currency (atributo de <i>py-flowcl.models.PaymentStatus</i>), 9	flowRefundOrder (atributo de <i>py-flowcl.models.RefundStatus</i>), 10
Customer (clase en <i>pyflowcl.models</i>), 5	flowTrxId (atributo de <i>pyflowcl.models.RefundRequest</i>), 9
customer_id (atributo de <i>py-flowcl.models.CollectObject</i>), 4	forward_days_after (atributo de <i>py-flowcl.models.BatchCollectRequest</i>), 3
CustomerChargeRequest (clase en <i>pyflowcl.models</i>), 6	forward_days_after (atributo de <i>py-flowcl.models.CollectRequest</i>), 5
customerId (atributo de <i>py-flowcl.models.BatchCollectRejectedRow</i>), 3	forward_days_after (atributo de <i>py-flowcl.models.PaymentRequestEmail</i>), 8
customerId (atributo de <i>py-flowcl.models.CollectRequest</i>), 5	forward_times (atributo de <i>py-flowcl.models.BatchCollectRequest</i>), 3
customerId (atributo de <i>pyflowcl.models.Customer</i>), 5	forward_times (atributo de <i>py-flowcl.models.CollectRequest</i>), 5
customerId (atributo de <i>py-flowcl.models.CustomerRegisterStatusResponse</i>), 7	forward_times (atributo de <i>py-flowcl.models.PaymentRequestEmail</i>), 8
customerId (atributo de <i>py-flowcl.models.CustomerRequest</i>), 7	from_dict() (método estático de <i>py-flowcl.models.BatchCollectRejectedRow</i>), 3
CustomerList (clase en <i>pyflowcl.models</i>), 6	from_dict() (método estático de <i>py-flowcl.models.BatchCollectRequest</i>), 3
CustomerRegisterResponse (clase en <i>py-flowcl.models</i>), 6	from_dict() (método estático de <i>py-flowcl.models.BatchCollectResponse</i>), 4
CustomerRegisterStatusResponse (clase en <i>py-flowcl.models</i>), 6	from_dict() (método estático de <i>py-flowcl.models.CollectObject</i>), 4
CustomerRequest (clase en <i>pyflowcl.models</i>), 7	from_dict() (método estático de <i>py-flowcl.models.CollectRequest</i>), 5
D	from_dict() (método estático de <i>py-flowcl.models.CollectResponse</i>), 5
data (atributo de <i>pyflowcl.models.CustomerList</i>), 6	from_dict() (método estático de <i>py-flowcl.models.Customer</i>), 6
data (atributo de <i>pyflowcl.models.PaymentList</i>), 7	from_dict() (método estático de <i>py-flowcl.models.CustomerChargeRequest</i>), 6
date (atributo de <i>pyflowcl.models.RefundStatus</i>), 10	from_dict() (método estático de <i>py-flowcl.models.CustomerList</i>), 6
E	from_dict() (método estático de <i>py-flowcl.models.CustomerRegisterResponse</i>), 6
email (atributo de <i>pyflowcl.models.Customer</i>), 5	from_dict() (método estático de <i>py-flowcl.models.CustomerRegisterStatusResponse</i>), 7
email (atributo de <i>pyflowcl.models.CustomerRequest</i>), 7	from_dict() (método estático de <i>py-flowcl.models.CustomerRequest</i>), 7
email (atributo de <i>pyflowcl.models.PaymentRequest</i>), 7	from_dict() (método estático de <i>py-flowcl.models.PaymentList</i>), 7
email (atributo de <i>py-flowcl.models.PaymentRequestEmail</i>), 8	from_dict() (método estático de <i>py-flowcl.models.PaymentRequest</i>), 7
errorCode (atributo de <i>py-flowcl.models.BatchCollectRejectedRow</i>), 3	
errorMsg (atributo de <i>py-flowcl.models.BatchCollectRejectedRow</i>), 3	
externalId (atributo de <i>pyflowcl.models.Customer</i>), 5	
externalId (atributo de <i>py-flowcl.models.CustomerRequest</i>), 7	
F	
fee (atributo de <i>pyflowcl.models.RefundStatus</i>), 10	

from_dict() (método estático de *py-flowcl.models.PaymentRequestEmail*), 8
 from_dict() (método estático de *py-flowcl.models.PaymentResponse*), 9
 from_dict() (método estático de *py-flowcl.models.PaymentStatus*), 9
 from_dict() (método estático de *py-flowcl.models.RefundRequest*), 9
 from_dict() (método estático de *py-flowcl.models.RefundStatus*), 10

G

GenericError, 7

get() (método de *pyflowcl.Clients.ApiClient*), 1
 getPayments() (en el módulo *pyflowcl.Payment*), 2
 getStatus() (en el módulo *pyflowcl.Payment*), 2
 getStatus() (en el módulo *pyflowcl.Refund*), 3
 getStatusByCommerceId() (en el módulo *py-flowcl.Payment*), 2
 getStatusByFlowOrder() (en el módulo *py-flowcl.Payment*), 2

H

hasMore (atributo de *pyflowcl.models.CustomerList*), 6
 hasMore (atributo de *pyflowcl.models.PaymentList*), 7

I

ignore_auto_charging (atributo de *py-flowcl.models.CollectRequest*), 5

L

last4CardDigits (atributo de *py-flowcl.models.Customer*), 6
 last4CardDigits (atributo de *py-flowcl.models.CustomerRegisterStatusResponse*), 7

M

módulo

pyflowcl, 10
pyflowcl.Clients, 1
pyflowcl.models, 3
pyflowcl.Payment, 2
pyflowcl.Refund, 3

make_signature() (método de *py-flowcl.Clients.ApiClient*), 2
 merchant_id (atributo de *py-flowcl.models.PaymentStatus*), 9
 merchantId (atributo de *py-flowcl.models.CollectRequest*), 5
 merchantId (atributo de *py-flowcl.models.PaymentRequest*), 7
 merchantId (atributo de *py-flowcl.models.PaymentRequestEmail*), 8

N

name (atributo de *pyflowcl.models.Customer*), 6
 name (atributo de *pyflowcl.models.CustomerRequest*), 7

O

optional (atributo de *pyflowcl.models.CollectObject*), 4
 optional (atributo de *py-flowcl.models.PaymentRequest*), 8
 optional (atributo de *py-flowcl.models.PaymentRequestEmail*), 8
 optional (atributo de *pyflowcl.models.PaymentStatus*), 9
 optionals (atributo de *pyflowcl.models.CollectRequest*), 5
 optionals (atributo de *py-flowcl.models.CustomerChargeRequest*), 6

P

parameter (atributo de *py-flowcl.models.BatchCollectRejectedRow*), 3
 pay_mode (atributo de *pyflowcl.models.Customer*), 6
 payer (atributo de *pyflowcl.models.PaymentStatus*), 9
 payment_result (atributo de *py-flowcl.models.CollectResponse*), 5
 payment_currency (atributo de *py-flowcl.models.PaymentRequest*), 8
 payment_currency (atributo de *py-flowcl.models.PaymentRequestEmail*), 8
 payment_data (atributo de *py-flowcl.models.PaymentStatus*), 9
 payment_method (atributo de *py-flowcl.models.CollectObject*), 4
 payment_method (atributo de *py-flowcl.models.PaymentRequest*), 8
 PaymentList (clase en *pyflowcl.models*), 7
 paymentMethod (atributo de *py-flowcl.models.CollectRequest*), 5
 PaymentRequest (clase en *pyflowcl.models*), 7
 PaymentRequestEmail (clase en *pyflowcl.models*), 8
 PaymentResponse (clase en *pyflowcl.models*), 8
 PaymentStatus (clase en *pyflowcl.models*), 9
 pending_info (atributo de *py-flowcl.models.PaymentStatus*), 9
 post() (método de *pyflowcl.Clients.ApiClient*), 2
 put() (método de *pyflowcl.Clients.ApiClient*), 2
 pyflowcl
 módulo, 10
 pyflowcl.Clients
 módulo, 1
 pyflowcl.models
 módulo, 3
 pyflowcl.Payment
 módulo, 2

pyflowcl.Refund
módulo, 3

R

receivedRows (atributo de py-flowcl.models.BatchCollectResponse), 4
receiverEmail (atributo de py-flowcl.models.RefundRequest), 9
refundCommerceOrder (atributo de py-flowcl.models.RefundRequest), 9
RefundRequest (clase en pyflowcl.models), 9
RefundStatus (clase en pyflowcl.models), 10
registerDate (atributo de pyflowcl.models.Customer), 6
rejectedRows (atributo de py-flowcl.models.BatchCollectResponse), 4
request_date (atributo de py-flowcl.models.PaymentStatus), 9
rowNumber (atributo de py-flowcl.models.BatchCollectRejectedRow), 3

S

s (atributo de pyflowcl.models.BatchCollectRequest), 3
s (atributo de pyflowcl.models.CollectRequest), 5
s (atributo de pyflowcl.models.CustomerChargeRequest), 6
s (atributo de pyflowcl.models.CustomerRequest), 7
s (atributo de pyflowcl.models.PaymentRequest), 8
s (atributo de pyflowcl.models.PaymentRequestEmail), 8
s (atributo de pyflowcl.models.RefundRequest), 9
status (atributo de pyflowcl.models.CollectResponse), 5
status (atributo de pyflowcl.models.Customer), 6
status (atributo de py-flowcl.models.CustomerRegisterStatusResponse), 7
status (atributo de pyflowcl.models.PaymentStatus), 9
status (atributo de pyflowcl.models.RefundStatus), 10
subject (atributo de pyflowcl.models.CollectObject), 4
subject (atributo de pyflowcl.models.CollectRequest), 5
subject (atributo de py-flowcl.models.CustomerChargeRequest), 6
subject (atributo de pyflowcl.models.PaymentRequest), 8
subject (atributo de py-flowcl.models.PaymentRequestEmail), 8
subject (atributo de pyflowcl.models.PaymentStatus), 9

T

timeout (atributo de py-flowcl.models.BatchCollectRequest), 3
timeout (atributo de pyflowcl.models.CollectRequest), 5

timeout (atributo de pyflowcl.models.PaymentRequest), 8

timeout (atributo de py-flowcl.models.PaymentRequestEmail), 8

token (atributo de py-flowcl.models.BatchCollectResponse), 4

token (atributo de pyflowcl.models.CollectResponse), 5

token (atributo de py-flowcl.models.CustomerRegisterResponse), 6

token (atributo de pyflowcl.models.PaymentResponse), 9

total (atributo de pyflowcl.models.CustomerList), 6

total (atributo de pyflowcl.models.PaymentList), 7

type (atributo de pyflowcl.models.CollectResponse), 5

U

url (atributo de pyflowcl.models.CollectResponse), 5

url (atributo de pyflowcl.models.CustomerRegisterResponse), 6

url (atributo de pyflowcl.models.PaymentResponse), 9

urlCallBack (atributo de py-flowcl.models.BatchCollectRequest), 4

urlCallBack (atributo de py-flowcl.models.RefundRequest), 9

urlConfirmation (atributo de py-flowcl.models.BatchCollectRequest), 4

urlConfirmation (atributo de py-flowcl.models.CollectRequest), 5

urlConfirmation (atributo de py-flowcl.models.PaymentRequest), 8

urlConfirmation (atributo de py-flowcl.models.PaymentRequestEmail), 8

urlReturn (atributo de py-flowcl.models.BatchCollectRequest), 4

urlReturn (atributo de pyflowcl.models.CollectRequest), 5

urlReturn (atributo de py-flowcl.models.PaymentRequest), 8

urlReturn (atributo de py-flowcl.models.PaymentRequestEmail), 8